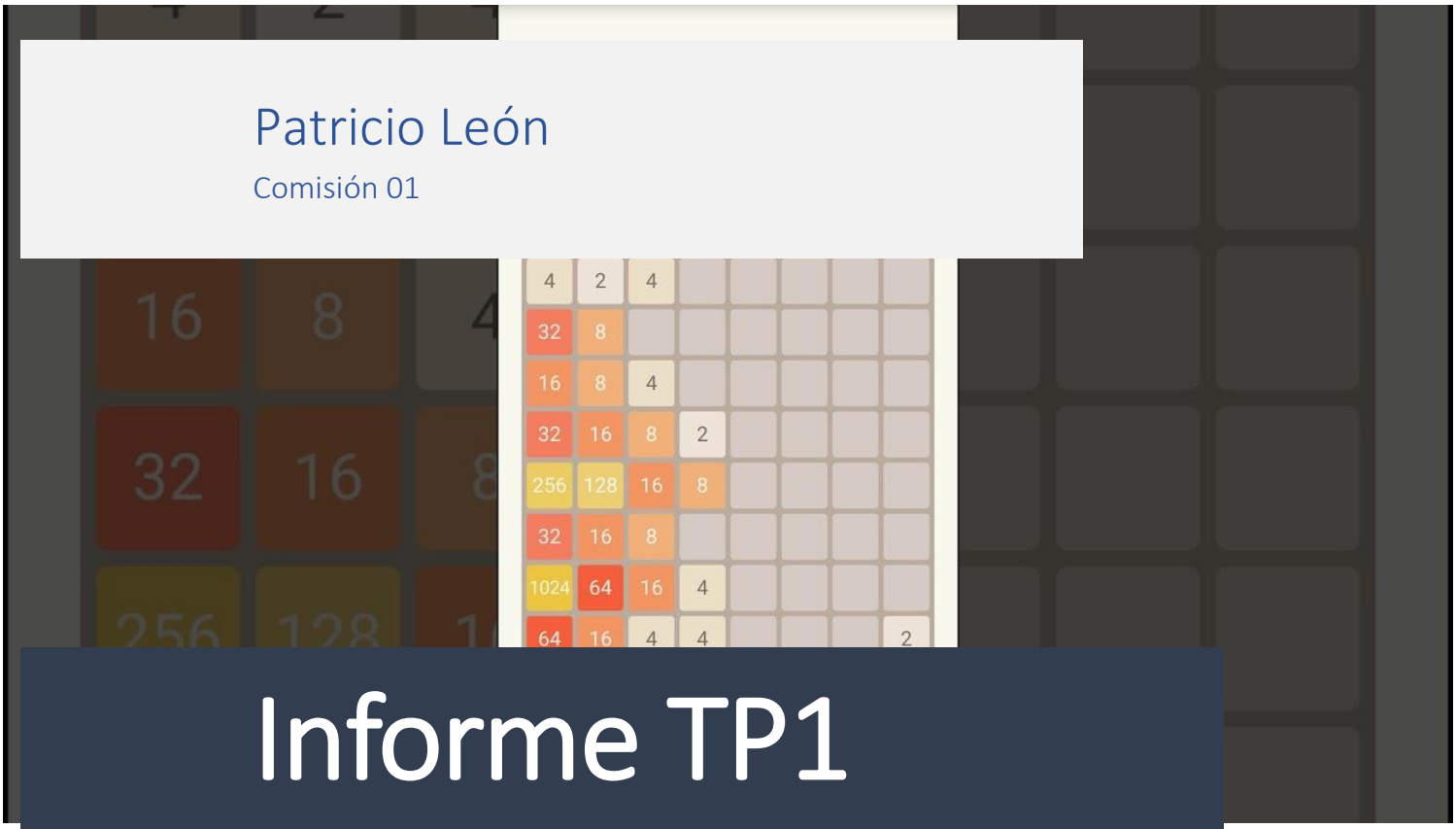
Patricio León
Comisión 01



Informe TP1

Juego 2048

Documentación del Proyecto: Implementación del Juego 2048 en Java

Introducción:

Este documento describe la implementación de una aplicación para jugar al juego "2048" en Java. El desarrollo del proyecto se ha dividido en dos partes principales: la lógica del juego y la interfaz gráfica del usuario. Cada parte está ubicada en un paquete separado para mantener una clara separación de responsabilidades y facilitar el mantenimiento del código.

Estructura del Proyecto

La estructura del proyecto es la siguiente:

```
PaquetesTP/
|-- Interfaz/
|   |-- interfazJuego.java
|
|-- Logica/
|   |-- Juego.java
```

Interfaz Gráfica

Clase: interfazJuego

Paquete: Interfaz

Descripción:

Esta clase maneja la interfaz gráfica del usuario utilizando JFrame y JPanel de Swing Captura las entradas del teclado para move los números y actualiza la cuadrícula en consecuencia.

Componentes y Decisiones Clave:

Método initialize():
Configura la ventana principal y añade los componentes necesarios.

Decisión: Utilizar un GridLayout para la cuadrícula de 4x4 y un BorderLayout para organizar la puntuación y la cuadrícula.

Método actualizarGrilla():
Actualiza la cuadrícula de la interfaz gráfica después de cada movimiento.

Decisión: Este método se llama después de cada movimiento del jugador para reflejar el estado actual de la cuadrícula y la puntuación.

Clase Interna KeyPressed:
Captura las teclas de flecha del teclado para mover los números en la dirección correspondiente.

Decisión: Utilizar un KeyEvent simplifica la captura de eventos del teclado y permite una interacción más natural con el juego.

Método colorNumeros(int value):
Devuelve el color de fondo de una celda basado en su valor.

Decisión: Asignar colores diferentes a los valores de las celdas mejora la experiencia visual del usuario y facilita l identificación rápida de los valores.

Método main(String[] args):
Inicia la interfaz gráfica del juego.

Decisión: Utilizar SwingUtilities.invokeLater para asegura que la interfaz gráfica se crea y se actualiza en el hilo de despacho de eventos de Swing.

Logica del juego

Clase: Juego

Paquete: Lógica

Descripción:

Esta clase contiene toda la lógica del juego, incluyendo la creación y actualización de la cuadrícula, la gestión de movimientos, la generación de nuevos números y la verificación de las condiciones de fin del juego.

Métodos y Decisiones Clave:

Constructor Juego():
Inicializa la cuadrícula 4x4 y añade dos números iniciales (2 o 4) en posiciones aleatorias.

Decisión: Comenzar con dos números en la cuadrícula sigue la convención del juego original.

Método addNumero():
Añade un nuevo número (2 o 4) en una posición aleatoria vacía de la cuadrícula.

Decisión: Se decidió utilizar una probabilidad del 90% para generar un 2 y del 10% para generar un 4, lo cual es consistente con el juego original.

Método move(Direction direction):
Gestiona el movimiento de los números en la dirección especificada (arriba, abajo, izquierda, derecha).

Decisión: Se separaron los movimientos en métodos específicos (moveArriba(), moveAbajo(), moveIzquierda(), moveDerecha()) para mantener el código limpio y modular.

Enumeración Direction:
Define las direcciones posibles de movimiento (ARRIBA, ABAJO, IZQUIERDA, DERECHA).

Métodos moveArriba(), moveAbajo(), moveIzquierda(), moveDerecha():
Implementan la lógica de los movimientos, incluyendo la combinación de números iguales y la actualización de la puntuación.

Decisión: Utilizar arreglos auxiliares para rastrear combinaciones y evitar combinaciones múltiples en un solo movimiento.

Método isGameOver():
Verifica si el juego ha terminado (sin movimientos posibles y sin celdas vacías).

Decisión: Este método se llama después de cada movimiento para determinar si el juego ha terminado y mostrar un mensaje de fin de juego si es necesario.

Método isGameOver():
Verifica si el juego ha terminado (sin movimientos posibles y sin celdas vacías).

Decisión: Este método se llama después de cada movimiento para determinar si el juego ha terminado y mostrar un mensaje de fin de juego si es necesario.

Decisión: Utilizar una enumeración mejora la legibilidad del código y evita errores con valores de cadena o enteros.

Separación de Responsabilidades:

Se dividió la lógica del juego y la interfaz gráfica en dos clases ubicadas en paquetes diferentes (logic y ui). Esta separación permite un mantenimiento más sencillo y una mayor claridad en el código.

Interacción del Usuario:

Se eligió capturar las teclas de flecha del teclado para mover los números, imitando la forma en que se juega el 2048 en dispositivos móviles y otros entornos. Esto proporciona una experiencia de usuario más intuitiva.

Colores de las Celdas:

Asignar colores específicos a los valores de las celdas ayuda a los jugadores a identificar rápidamente las diferentes celdas y a tomar decisiones estratégicas más fácilmente.

Modularidad:

La lógica de los movimientos se dividió en métodos específicos para cada dirección (moveUp(), moveDown(), moveLeft(), moveRight()), lo que mejora la modularidad y la legibilidad del código.

Conclusión:

La implementación del juego 2048 en Java se logró mediante una cuidadosa separación de la lógica del juego y la interfaz gráfica, así como mediante decisiones de diseño que mejoran la experiencia del usuario y la mantenibilidad del código. Esta estructura modular y clara facilita futuras mejoras y expansiones del juego.